UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
10/720,960	11/24/2003	Richard D. Dettinger	ROC920030277US1	5201

46797 7590 03/03/2011
IBM CORPORATION, INTELLECTUAL PROPERTY LAW
DEPT 917, BLDG. 006-1
3605 HIGHWAY 52 NORTH
ROCHESTER, MN 55901-7829

EXAMINER	
DWIVEDI, MAHESH H	

ART UNIT	PAPER NUMBER
2168	

MAIL DATE	DELIVERY MODE
03/03/2011	PAPER

Please find below and/or attached an Office communication concerning this application or proceeding.

The time period for reply, if any, is set in the attached communication.

BEFORE THE BOARD OF PATENT APPEALS AND INTERFERENCES

Application Number: 10/720,960
Filing Date: November 24, 2003
Appellant(s): DETTINGER ET AL.

Geo McClellan (Reg # 44, 227)
For Appellant

EXAMINER'S ANSWER

This is in response to the appeal brief filed 01/03/2011 appealing from the Office action mailed 08/02/2010.

(1) Real Party in Interest

A statement identifying by name the real party in interest is contained in the brief.

(2) Related Appeals and Interferences

The examiner is not aware of any related appeals, interferences, or judicial proceedings which will directly affect or be directly affected by or have a bearing on the Board's decision in the pending appeal.

(3) Status of Claims

The statement of the status of claims in the brief is correct.

(4) Status of Amendments After Final

No amendment after final has been filed.

(5) Summary of Claimed Subject Matter

The summary of claimed subject matter contained in the brief is correct.

(6) Grounds of Rejection to be Reviewed on Appeal

The appellant's statement of the grounds of rejection to be reviewed on appeal is correct.

(7) Claims Appendix

The copy of the appealed claims contained in the Appendix to the brief is correct.

(8) Evidence Relied Upon

6,453,353	Win et al.	09/17/2002
2003/0158839	Faybishenko et al.	08/21/2003
6,968,509	Chang et al.	11/22/2005

(9) Grounds of Rejection

The following ground(s) of rejection are applicable to the appealed claims:

Claim Rejections - 35 USC § 103

1. The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

(a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the manner in which the invention was made.

2. This application currently names joint inventors. In considering patentability of the claims under 35 U.S.C. 103(a), the examiner presumes that the subject matter of the various claims was commonly owned at the time any inventions covered therein were made absent any evidence to the contrary. Applicant is advised of the obligation under 37 CFR 1.56 to point out the inventor and invention dates of each claim that was not commonly owned at the time a later invention was made in order for the examiner to consider the applicability of 35 U.S.C. 103(c) and potential 35 U.S.C. 102(e), (f) or (g) prior art under 35 U.S.C. 103(a).

3. Claims 10, 14-16, 20-21, and 27-32 are rejected under 35 U.S.C. 103(a) as being unpatentable over **Win et al.** (U.S. Patent 6,453,353) in view of **Faybishenko et al.** (U.S. PGPUB 2003/0158839), and further in view of **Chang et al.** (U.S. Patent 6,968,509)

4. Regarding claim 10, **Win** teaches a method comprising:

A) assigning metadata requirements to functional modules that operate on data stored in, or functional modules that generate results that are stored in, a database (Abstract, Column 5, lines 44-46, Column 6, lines 10-16, lines 41-65);

B) wherein the assigned metadata requirements specify conditions required for successful execution of the functional module (Abstract, Column 5, lines 44-46, Column 6, lines 10-16, lines 41-65);

C) wherein at least one condition defines at least one user role required for successful; execution of the functional module (Abstract, Column 5, lines 44-46, Column 6, lines 10-16, lines 41-65);

H) obtaining a list of functional modules that are accessible from within the application used during the query session (Abstract, Column 6, lines 10-16, lines 41-65);
I) identifying a limited subset of the functional modules in the list that will successfully execute, by comparing the collected runtime metadata with the assigned metadata requirements (Abstract, Column 6, lines 10-16, lines 41-65); and
J) providing an interface presenting the user with the identified limited subset of functional modules that will successfully execute (Abstract, Column 6, lines 10-16, lines 41-65).

The examiner notes that **Win** teaches **"assigning metadata requirements to functional modules that operate on data stored in, or functional modules that generate results that are stored in, a database"** as "Roles determine what resources a User can access. Further, each role may require a set of information that is available in resources" (Column 5, lines 44-46) and "When the user selects a resource, a browser sends an open URL request and cookie to a Protected Web Server. A Protected Web Server is a web server with resources protected by the Runtime Module decrypts information in the cookie and uses it to verify that he user is authorized to access the resource" (Column 6, lines 58-64). The examiner further notes that **Win** teaches **"wherein the assigned metadata requirements specify conditions required for successful execution of the functional module"** as "the runtime module on the protected server receives the login request and intercepts all other request by the client to use a resource" (Abstract), "If the name and password are correct, the Authentication Client Module reads the user's roles from the Registry server" (Column 6, lines 44-46), and "a personalized menu is an HTML page containing a list of authorized Resources" (Column 6, lines 13-14). The examiner further notes that **Win** teaches **"wherein at least one condition defines at least one user role required for successful; execution of the functional module"** as "the runtime module on the protected server receives the login request and intercepts all other request by the client to use a resource" (Abstract), "If the name and password are correct, the Authentication Client Module reads the user's roles from the Registry server" (Column 6, lines 44-46), and "a personalized menu is an HTML page containing a list of authorized Resources"

(Column 6, lines 13-14). The examiner further notes that **Win** teaches **"obtaining a list of functional modules that are accessible from within the application used during the query session"** as "When the user selects a resource, a browser sends an open URL request and cookie to a Protected Web Server. A Protected Web Server is a web server with resources protected by the Runtime Module decrypts information in the cookie and uses it to verify that he user is authorized to access the resource" (Column 6, lines 58-64). The examiner further notes that **Win** teaches **"identifying a limited subset of the functional modules in the list that will successfully execute, by comparing the collected runtime metadata with the assigned metadata requirements"** as "a Personalized Menu is an HTML page containing a list of authorized resources. The Personalized Menus displays only Resources to which the User has access" (Column 6, lines 12-15) and "When the user selects a resource, a browser sends an open URL request and cookie to a Protected Web Server. A Protected Web Server is a web server with resources protected by the Runtime Module decrypts information in the cookie and uses it to verify that he user is authorized to access the resource" (Column 6, lines 58-64). The examiner further notes that **Win** teaches **"providing an interface presenting the user with the identified limited subset of functional modules that will successfully execute"** as "a Personalized Menu is an HTML page containing a list of authorized resources. The Personalized Menus displays only Resources to which the User has access" (Column 6, lines 12-15).

Win does not explicitly teach:

D) collecting runtime metadata relating to one or more result fields in a query statement;

E) wherein the one or more result fields specify one or more data fields for which data is requested to be returned upon execution of the query statement;

F) wherein the runtime metadata is collected after composition of the query statement.

Faybishenko, however, teaches **"collecting runtime metadata relating to one or more result fields of a query"** as "In one embodiment a QRP adapter may monitor or log queries, results, number of hits, searches, results, etc. or generally the information passing through the QRP adapter. In one embodiment, a user interface may be provided through which providers may view the results of searches and hits performed by consumers--e.g. how many searches resulted in their entry being returned, how many users clicked through, etc. In one embodiment, a user interface may be provided through which providers may monitor and/or control the number of queries sent to them and also to throttle traffic (e.g. turn it off) if necessary. In some embodiments, a QRP interface may be able to access a registration file, for example to read at least part of the registration document or to write to replace or to add to at least part of the registration document" (Paragraph 112), **"wherein the one or more result fields specify one or more data fields for which data is requested to be returned upon execution of the query statement"** as "In one embodiment a QRP adapter may monitor or log queries, results, number of hits, searches, results, etc. or generally the information passing through the QRP adapter. In one embodiment, a user interface may be provided through which providers may view the results of searches and hits performed by consumers--e.g. how many searches resulted in their entry being returned, how many users clicked through, etc. In one embodiment, a user interface may be provided through which providers may monitor and/or control the number of queries sent to them and also to throttle traffic (e.g. turn it off) if necessary. In some embodiments, a QRP interface may be able to access a registration file, for example to read at least part of the registration document or to write to replace or to add to at least part of the registration document" (Paragraph 112), and **"wherein the runtime metadata is collected after composition of the query statement"** as "In one embodiment a QRP adapter may monitor or log queries, results, number of hits, searches, results, etc. or generally the information passing through the QRP adapter. In one embodiment, a user interface may be provided through which providers may view the results of searches and hits performed by consumers--e.g. how many searches resulted in their entry being returned, how many users clicked through, etc. In one embodiment, a user interface may be provided through which providers may monitor and/or control the number of queries sent to them and also to throttle traffic (e.g. turn it off) if necessary. In some embodiments, a QRP interface may be able to access a registration file, for example to read at least part of the registration document or to write to replace or to add to at least part of the registration document" (Paragraph 112).

The examiner further notes that logging an entire query teaches the claimed result fields because the result fields are encompassed within that query.

It would have been obvious to one of ordinary skill in the art at the time the invention was made to combine the teachings of the cited references because teaching **Faybishenko's** would have allowed **Win's** to provide a method to allow for more control over content provided over the internet to providers, as noted by **Faybishenko** (Paragraph 6).

Win and **Faybishenko** do not explicitly teach:

G) wherein the runtime metadata is collected before the query statement is submitted for execution.

Chang, however, teaches **"wherein the runtime metadata is collected before the query statement is submitted for execution"** as "The logical operations of the executable begin at signal operation 202 where the CPU 104 awaits a user activity signal that is provided from the logical operations of FIG. 3. The user activity signal specifies when a particular key on the keyboard has been pressed by the user and when a particular mouse button has been clicked by the user. Query operation 204 detects whether the signal has been received. If not, then signal operation 202 continues to await the user activity signal. If the signal of user activity has been received, then query operation 206 detects whether the signal specifies a mouse click as opposed to a keyboard type. When a keyboard type is detected instead of a mouse click, then record operation 208 records the keyboard type as a user-driven event. As discussed, recording the user-driven event may involve one or more techniques, such as displaying a textual description of the keyboard type on the display screen, saving a description to an electronic file, and/or printing the textual description via a printer. Each character that is typed may be placed on the same line of the textual description as shown in the screenshots until the typed key is an <Enter> key or unless there is a control key such as <Backspace> or <Ctrl>+"A". After each keystroke is recorded, operational flow returns to signal operation 202" (Column 5, lines 4-26).

The examiner notes that **Chang** records, saves, and displays all concurrent typed keyboard strokes by a user. Because **Chang** records the typed keyboard strokes as a user is typing them, then, as a result, the typed keywords are saved before a "submission of execution".

It would have been obvious to one of ordinary skill in the art at the time the invention was made to combine the teachings of the cited references because teaching **Chang's** would have allowed **Win's** and **Faybishenko's** to provide a method that adequately records user events, as noted by **Chang** (Column 1, lines 49-50).

Regarding claim 14, **Win** further teaches a method comprising:
A) wherein obtaining metadata associated with the functional module comprises examining a signature validation (Column 6, lines 1-3, Column 14, lines 34-43).

The examiner notes that **Win** teaches **"wherein obtaining metadata associated with the functional module comprises examining a signature validation"** as "users may log in either with a digital certificate or by opening a login page URL with a web browser and entering a name and password" (Column 6, lines 1-3).

Regarding claim 15, **Win** further teaches a method comprising:
A) wherein the metadata associated with at least one of the functional modules comprises at least one of: one or more input parameters required for successful execution of the functional module, one or more output parameters required for successful execution of the functional module, and a security credential required to execute the functional module (Abstract, Column 6, lines 10-16, lines 41-65).

The examiner notes that **Win** teaches **"wherein the metadata associated with at least one of the functional modules comprises at least one of one or more input parameters required for successful execution of the functional module; one or more output parameters required for successful execution of the functional module; and a credential of a user authorized to execute the functional module"** as “The Authentication Client Module and Access Menu Module authenticates a user by verifying the name and password with the Registry Server 108” (Column 6, lines 42-44).

Regarding claim 16, **Win** does not explicitly teach a method comprising:

A) wherein at least one of the functional modules analyzes query results.

Faybishenko, however, teaches **“wherein at least one of the functional modules analyzes query results”** as “In one embodiment a QRP adapter may monitor or log queries, results, number of hits, searches, results, etc. or generally the information passing through the QRP adapter. In one embodiment, a user interface may be provided through which providers may view the results of searches and hits performed by consumers--e.g. how many searches resulted in their entry being returned, how many users clicked through, etc. In one embodiment, a user interface may be provided through which providers may monitor and/or control the number of queries sent to them and also to throttle traffic (e.g. turn it off) if necessary. In some embodiments, a QRP interface may be able to access a registration file, for example to read at least part of the registration document or to write to replace or to add to at least part of the registration document” (Paragraph 112).

The examiner further notes that storing the click history of users teaches the claimed analyzing of query results.

It would have been obvious to one of ordinary skill in the art at the time the invention was made to combine the teachings of the cited references because teaching **Faybishenko’s** would have allowed **Win’s** to provide a method to allow for more control over content provided over the internet to providers, as noted by **Faybishenko** (Paragraph 6).

Regarding claim 20, **Win** teaches computer readable storage medium comprising:

A) assigning metadata requirements to functional modules that operate on data stored in, or functional modules that generate results that are stored in, a database (Abstract, Column 5, lines 44-46, Column 6, lines 10-16, lines 41-65);

B) wherein the assigned metadata requirements specify conditions required for successful execution of the functional module (Abstract, Column 5, lines 44-46, Column 6, lines 10-16, lines 41-65);

C) wherein at least one condition defines at least one user role required for successful execution of the functional module (Abstract, Column 5, lines 44-46, Column 6, lines 10-16, lines 41-65);

H) obtaining a list of functional modules accessible from within the application (Abstract, Column 6, lines 10-16, lines 41-65);

I) identifying a limited subset of the functional modules that will successfully execute, by comparing the collected runtime metadata with the assigned metadata requirements (Abstract, Column 6, lines 10-16, lines 41-65); and

J) providing an interface presenting the user with the identified limited subset of functional modules that will successfully execute (Abstract, Column 6, lines 10-16, lines 41-65).

The examiner notes that **Win** teaches **"assigning metadata requirements to functional modules that operate on data stored in, or functional modules that generate results that are stored in, a database"** as "Roles determine what resources a User can access. Further, each role may require a set of information that is available in resources" (Column 5, lines 44-46) and "When the user selects a resource, a browser sends an open URL request and cookie to a Protected Web Server. A Protected Web Server is a web server with resources protected by the Runtime Module decrypts information in the cookie and uses it to verify that he user is authorized to access the resource" (Column 6, lines 58-64). The examiner further notes that **Win** teaches **"wherein the assigned metadata requirements specify conditions required for successful execution of the functional module"** as "the runtime module on the protected server receives the login request and intercepts all other request by the client to use a resource" (Abstract), "If the name and password are correct, the Authentication Client Module reads the user's roles from the Registry server" (Column 6, lines 44-46), and "a personalized menu is an HTML page containing a list of authorized Resources" (Column 6, lines 13-14). The examiner further notes that **Win** teaches **“wherein at least one condition defines at least one user role required for successful; execution of the functional module”** as “the runtime module on the protected server receives the login request and intercepts all other request by the client to use a resource” (Abstract), “If the name and password are correct, the Authentication Client Module reads the user’s roles from the Registry server” (Column 6, lines 44-46), and “a personalized menu is an HTML page containing a list of authorized Resources” (Column 6, lines 13-14). The examiner further notes that **Win** teaches **“obtaining a list of functional modules accessible from within the application”** as “a list of authorized resources” (Column 6, lines 13-14) and “When the user selects a resource, a browser sends an open URL request and cookie to a Protected Web Server. A Protected Web Server is a web server with resources protected by the Runtime Module decrypts information in the cookie and uses it to verify that he user is authorized to access the resource” (Column 6, lines 58-64). The examiner further notes that **Win** teaches **“identifying a limited subset of the functional modules that will successfully execute, by comparing the collected runtime metadata with the assigned metadata requirements”** as “a Personalized Menu is an HTML page containing a list of authorized resources. The Personalized Menus displays only Resources to which the User has access” (Column 6, lines 12-15) and “When the user selects a resource, a browser sends an open URL request and cookie to a Protected Web Server. A Protected Web Server is a web server with resources protected by the Runtime Module decrypts information in the cookie and uses it to verify that he user is authorized to access the resource” (Column 6, lines 58-64). The examiner further notes that **Win** teaches **“providing an interface presenting the user with the identified limited subset of functional modules that will successfully execute”** as “a Personalized Menu is an HTML page containing a list of authorized resources. The Personalized Menus displays only Resources to which the User has access” (Column 6, lines 12-15).

Win does not explicitly teach:

D) collecting runtime metadata relating to one or more result fields in a query statement;

E) wherein the one or more result fields specify one or more data fields for which data is requested to be returned upon execution of the query statement;

F) wherein the runtime metadata is collected after composition of the query statement.

Faybishenko, however, teaches **"collecting runtime metadata relating to one or more result fields in a query statement"** as "In one embodiment a QRP adapter may monitor or log queries, results, number of hits, searches, results, etc. or generally the information passing through the QRP adapter. In one embodiment, a user interface may be provided through which providers may view the results of searches and hits performed by consumers--e.g. how many searches resulted in their entry being returned, how many users clicked through, etc. In one embodiment, a user interface may be provided through which providers may monitor and/or control the number of queries sent to them and also to throttle traffic (e.g. turn it off) if necessary. In some embodiments, a QRP interface may be able to access a registration file, for example to read at least part of the registration document or to write to replace or to add to at least part of the registration document" (Paragraph 112), **"wherein the one or more result fields specify one or more data fields for which data is requested to be returned upon execution of the query statement"** as "In one embodiment a QRP adapter may monitor or log queries, results, number of hits, searches, results, etc. or generally the information passing through the QRP adapter. In one embodiment, a user interface may be provided through which providers may view the results of searches and hits performed by consumers--e.g. how many searches resulted in their entry being returned, how many users clicked through, etc. In one embodiment, a user interface may be provided through which providers may monitor and/or control the number of queries sent to them and also to throttle traffic (e.g. turn it off) if necessary. In some embodiments, a QRP interface may be able to access a registration file, for example to read at least part of the registration document or to write to replace or to add to at least part of the registration document" (Paragraph 112), and **"wherein the runtime metadata is collected after composition of the query statement"** as "In one embodiment a QRP adapter may monitor or log queries, results, number of hits, searches, results, etc. or generally the information passing through the QRP adapter. In one embodiment, a user interface may be provided through which providers may view the results of searches and hits performed by consumers--e.g. how many searches resulted in their entry being returned, how many users clicked through, etc. In one embodiment, a user interface may be provided through which providers may monitor and/or control the number of queries sent to them and also to throttle traffic (e.g. turn it off) if necessary. In some embodiments, a QRP interface may be able to access a registration file, for example to read at least part of the registration document or to write to replace or to add to at least part of the registration document" (Paragraph 112).

The examiner further notes that logging an entire query teaches the claimed result fields because the result fields are encompassed within that query.

It would have been obvious to one of ordinary skill in the art at the time the invention was made to combine the teachings of the cited references because teaching **Faybishenko's** would have allowed **Win's** to provide a method to allow for more control over content provided over the internet to providers, as noted by **Faybishenko** (Paragraph 6).

Win and **Faybishenko** do not explicitly teach:

G) wherein the runtime metadata is collected before the query statement is submitted for execution.

Chang, however, teaches **"wherein the runtime metadata is collected before the query statement is submitted for execution"** as "The logical operations of the executable begin at signal operation 202 where the CPU 104 awaits a user activity signal that is provided from the logical operations of FIG. 3. The user activity signal specifies when a particular key on the keyboard has been pressed by the user and when a particular mouse button has been clicked by the user. Query operation 204 detects whether the signal has been received. If not, then signal operation 202 continues to await the user activity signal. If the signal of user activity has been received, then query operation 206 detects whether the signal specifies a mouse click as opposed to a keyboard type. When a keyboard type is detected instead of a mouse click, then record operation 208 records the keyboard type as a user-driven event. As discussed, recording the user-driven event may involve one or more techniques, such as displaying a textual description of the keyboard type on the display screen, saving a description to an electronic file, and/or printing the textual description via a printer. Each character that is typed may be placed on the same line of the textual description as shown in the screenshots until the typed key is an <Enter> key or unless there is a control key such as <Backspace> or <Ctrl>+"A". After each keystroke is recorded, operational flow returns to signal operation 202" (Column 5, lines 4-26).

The examiner notes that **Chang** records, saves, and displays all concurrent typed keyboard strokes by a user. Because **Chang** records the typed keyboard strokes as a user is typing them, then, as a result, the typed keywords are saved before a "submission of execution".

It would have been obvious to one of ordinary skill in the art at the time the invention was made to combine the teachings of the cited references because teaching **Chang's** would have allowed **Win's** and **Faybishenko's** to provide a method that adequately records user events, as noted by **Chang** (Column 1, lines 49-50).

Regarding claim 21, **Win** does not explicitly teach a computer readable storage medium comprising:

A) wherein the application is a query building application.

Faybishenko, however, teaches **"wherein the application is a query building application"** as "In one embodiment a QRP adapter may monitor or log queries, results, number of hits, searches, results, etc. or generally the information passing through the QRP adapter. In one embodiment, a user interface may be provided through which providers may view the results of searches and hits performed by consumers--e.g. how many searches resulted in their entry being returned, how many users clicked through, etc. In one embodiment, a user interface may be provided through which providers may monitor and/or control the number of queries sent to them and also to throttle traffic (e.g. turn it off) if necessary. In some embodiments, a QRP interface may be able to access a registration file, for example to read at least part of the registration document or to write to replace or to add to at least part of the registration document" (Paragraph 112)

It would have been obvious to one of ordinary skill in the art at the time the invention was made to combine the teachings of the cited references because teaching **Faybishenko's** would have allowed **Win's** to provide a method to allow for more control over content provided over the internet to providers, as noted by **Faybishenko** (Paragraph 6).

Regarding claim 27, **Win** teaches a data processing system comprising:
A) a data repository (Column 5, lines 13-15);
B) a plurality of functional modules, each having associated metadata requirements that specify conditions required for successful execution of the functional modules (Abstract, Column 6, lines 10-16, lines 41-65);
C) wherein at least one condition defines at least one user role required for successful execution of the functional modules (Abstract, Column 5, lines 44-46, Column 6, lines 10-16, lines 41-65);
D) an application from which the functional modules are accessible (Abstract, Column 6, lines 10-16, lines 41-65);
G) present to a user a limited subset of the functional modules that will successfully execute, as determined by the application based on the collected runtime metadata and the metadata requirements associated with the functional modules (Abstract, Column 5, lines 66-67-Column 6, lines 1-16).

The examiner notes that Win teaches **"a data repository"** as "The system 2 enables organizations to register information sources or Resources and register Users of the information in a central repository" (Column 5, lines 13-15). The examiner further notes that **Win** teaches **"a plurality of functional modules, each having associated metadata requirements that specify conditions required for successful execution of the functional modules"** as "a list of authorized resources" (Column 6, lines 13-14). The examiner further notes that **Win** teaches **"an application from which the functional modules are accessible"** as "a personalized menu is an HTML page containing a list of authorized Resources" (Column 6, lines 13-14). The examiner further notes that **Win** teaches **"wherein at least one condition defines at least one**

user role required for successful execution of the functional modules" as "the runtime module on the protected server receives the login request and intercepts all other request by the client to use a resource" (Abstract), "If the name and password are correct, the Authentication Client Module reads the user's roles from the Registry server" (Column 6, lines 44-46), and "a personalized menu is an HTML page containing a list of authorized Resources" (Column 6, lines 13-14). The examiner further notes that **Win** teaches **"present to a user a limited subset of the functional modules that will successfully execute, as determined by the application based on the collected runtime metadata and the metadata requirements associated with the functional modules"** as "the runtime module on the protected server receives the login request and intercepts all other request by the client to use a resource" (Abstract), "If the name and password are correct, the Authentication Client Module reads the user's roles from the Registry server" (Column 6, lines 44-46), and "a personalized menu is an HTML page containing a list of authorized Resources" (Column 6, lines 13-14).

Win does not explicitly teach:

F) wherein the one or more result fields specify one or more data fields for which data is requested to be returned upon execution of the query statement;

Faybishenko, however, teaches **"wherein the one or more result fields specify one or more data fields for which data is requested to be returned upon execution of the query statement"** as "In one embodiment a QRP adapter may monitor or log queries, results, number of hits, searches, results, etc. or generally the information passing through the QRP adapter. In one embodiment, a user interface may be provided through which providers may view the results of searches and hits performed by consumers--e.g. how many searches resulted in their entry being returned, how many users clicked through, etc. In one embodiment, a user interface may be provided through which providers may monitor and/or control the number of queries sent to them and also to throttle traffic (e.g. turn it off) if necessary. In some embodiments, a QRP interface may be able to access a registration file, for example to read at least part of the registration document or to write to replace or to add to at least part of the registration document" (Paragraph 112).

The examiner further notes that logging an entire query teaches the claimed result fields because the result fields are encompassed within that query.

It would have been obvious to one of ordinary skill in the art at the time the invention was made to combine the teachings of the cited references because teaching **Faybishenko's** would have allowed **Win's** to provide a method to allow for more control over content provided over the internet to providers, as noted by **Faybishenko** (Paragraph 6).

Win and **Faybishenko** do not explicitly teach:

E) wherein the application is configured to: after composition of a query statement, but before the query statement is submitted for execution, collect runtime metadata related to one or more result fields in the query statement.

Chang, however, teaches **"wherein the application is configured to: after composition of a query statement, but before the query statement is submitted for execution, collect runtime metadata related to one or more result fields in the query statement"** as "The logical operations of the executable begin at signal operation 202 where the CPU 104 awaits a user activity signal that is provided from the logical operations of FIG. 3. The user activity signal specifies when a particular key on the keyboard has been pressed by the user and when a particular mouse button has been clicked by the user. Query operation 204 detects whether the signal has been received. If not, then signal operation 202 continues to await the user activity signal. If the signal of user activity has been received, then query operation 206 detects whether the signal specifies a mouse click as opposed to a keyboard type. When a keyboard type is detected instead of a mouse click, then record operation 208 records the keyboard type as a user-driven event. As discussed, recording the user-driven event may involve one or more techniques, such as displaying a textual description of the keyboard type on the display screen, saving a description to an electronic file, and/or printing the textual description via a printer. Each character that is typed may be placed on the same line of the textual description as shown in the screenshots until the typed key is an <Enter> key or unless there is a control key such as <Backspace> or

<Ctrl>+"A". After each keystroke is recorded, operational flow returns to signal operation 202" (Column 5, lines 4-26).

The examiner notes that **Chang** records, saves, and displays all concurrent typed keyboard strokes by a user. Because **Chang** records the typed keyboard strokes as a user is typing them, then, as a result, the typed keywords are saved before a "submission of execution".

It would have been obvious to one of ordinary skill in the art at the time the invention was made to combine the teachings of the cited references because teaching **Chang's** would have allowed **Win's** and **Faybishenko's** to provide a method that adequately records user events, as noted by **Chang** (Column 1, lines 49-50).

Regarding claim 28, **Win** does not explicitly teach a data processing system comprising:
A) wherein the data repository comprises XML data structures used to store runtime metadata.

Faybishenko, however, teaches **"wherein the data repository comprises XML data structures used to store runtime metadata"** as "In some embodiments, users and end applications (consumers 140) may present queries to a distributed information discovery network as arbitrary XML. Schema selection may be performed by HTTP header specification, in some embodiments. In one embodiment, queries presented by consumers 140 may adhere to specific queryspaces. In some embodiments, queries may be routed to the appropriate provider 120 by sending requests (e.g. XML requests) over HTTP" (Paragraph 54).

It would have been obvious to one of ordinary skill in the art at the time the invention was made to combine the teachings of the cited references because teaching **Faybishenko's** would have allowed **Win's** to provide a method to allow for more control over content provided over the internet to providers, as noted by **Faybishenko** (Paragraph 6).

Regarding claim 29, **Win** further teaches a data processing system comprising:

A) wherein the data repository comprises relational database tables used to store runtime metadata (Column 5, lines 13-15, Column 7, lines 1-6).

The examiner notes that **Win** teaches **"wherein the data repository comprises relational database tables used to store runtime metadata"** as "The Registry Repository is structured as a database. For example, the Registry Repository may be an SQL Server relational database management system, the Oracle 7® database, etc." (Column 7, lines 1-6). The examiner further notes that it is common knowledge that relational databases store data in tables.

Regarding claim 30, **Win** does not explicitly teach a method comprising:
A) wherein the runtime metadata relating to one or more result fields in the query statement comprises one or more of: a result field name; and a data type.

Faybishenko, however, teaches **"wherein the runtime metadata relating to one or more result fields in the query statement comprises one or more of: a result field name; and a data type"** as "In one embodiment a QRP adapter may monitor or log queries, results, number of hits, searches, results, etc. or generally the information passing through the QRP adapter. In one embodiment, a user interface may be provided through which providers may view the results of searches and hits performed by consumers--e.g. how many searches resulted in their entry being returned, how many users clicked through, etc. In one embodiment, a user interface may be provided through which providers may monitor and/or control the number of queries sent to them and also to throttle traffic (e.g. turn it off) if necessary. In some embodiments, a QRP interface may be able to access a registration file, for example to read at least part of the registration document or to write to replace or to add to at least part of the registration document" (Paragraph 112).

The examiner further notes that logging an entire query teaches the claimed result fields because the result fields are encompassed within that query.

It would have been obvious to one of ordinary skill in the art at the time the invention was made to combine the teachings of the cited references because teaching **Faybishenko's** would have allowed **Win's** to provide a method to allow for more control over content provided over the internet to providers, as noted by **Faybishenko** (Paragraph 6).

Regarding claim 31, **Win** does not explicitly teach a computer readable storage medium comprising:

A) wherein the runtime metadata relating to one or more result fields in the query statement comprises one or more of: a result field name; and a data type for the result field.

Faybishenko, however, teaches **"wherein the runtime metadata relating to one or more result fields in the query statement comprises one or more of: a result field name; and a data type for the result field"** as "In one embodiment a QRP adapter may monitor or log queries, results, number of hits, searches, results, etc. or generally the information passing through the QRP adapter. In one embodiment, a user interface may be provided through which providers may view the results of searches and hits performed by consumers--e.g. how many searches resulted in their entry being returned, how many users clicked through, etc. In one embodiment, a user interface may be provided through which providers may monitor and/or control the number of queries sent to them and also to throttle traffic (e.g. turn it off) if necessary. In some embodiments, a QRP interface may be able to access a registration file, for example to read at least part of the registration document or to write to replace or to add to at least part of the registration document" (Paragraph 112).

The examiner further notes that logging an entire query teaches the claimed result fields because the result fields are encompassed within that query.

It would have been obvious to one of ordinary skill in the art at the time the invention was made to combine the teachings of the cited references because teaching **Faybishenko's** would have allowed **Win's** to provide a method to allow for more control over content provided over the internet to providers, as noted by **Faybishenko** (Paragraph 6).

Regarding claim 32, **Win** does not explicitly teach a data processing system comprising:

A) wherein the runtime metadata relating to one or more result fields in the query statement comprises one or more of: a result field name; and a data type for the result field.

Faybishenko, however, teaches **"wherein the runtime metadata relating to one or more result fields in the query statement comprises one or more of: a result field name; and a data type for the result field"** as "In one embodiment a QRP adapter may monitor or log queries, results, number of hits, searches, results, etc. or generally the information passing through the QRP adapter. In one embodiment, a user interface may be provided through which providers may view the results of searches and hits performed by consumers--e.g. how many searches resulted in their entry being returned, how many users clicked through, etc. In one embodiment, a user interface may be provided through which providers may monitor and/or control the number of queries sent to them and also to throttle traffic (e.g. turn it off) if necessary. In some embodiments, a QRP interface may be able to access a registration file, for example to read at least part of the registration document or to write to replace or to add to at least part of the registration document" (Paragraph 112).

The examiner further notes that logging an entire query teaches the claimed result fields because the result fields are encompassed within that query.

It would have been obvious to one of ordinary skill in the art at the time the invention was made to combine the teachings of the cited references because teaching **Faybishenko's** would have allowed **Win's** to provide a method to allow for more control over content provided over the internet to providers, as noted by **Faybishenko** (Paragraph 6).

(10) Response to Argument

A. Claims 10, 14-16, 20-21, and 27-32 are rejected under 35 U.S.C. 103(a) as being unpatentable over **Win et al.** (U.S. Patent 6,453,353) in view of **Faybishenko et al.** (U.S. PGPUB 2003/0158839), and further in view of **Chang et al.** (U.S. Patent 6,968,509).

1. Independent Claims 10, 20, and 27:

Arguments (1):

I) Regarding Independent Claims 10, 20, and 27, Appellant argues that "Applicants submit that Dogl does not disclose a method of providing a user access to functional modules from within a query application that includes "collecting runtime metadata relating to one or more result fields in a query statement, wherein the one or more result fields specify one or more data fields for which data is requested to be returned upon execution of the query statement".

However, appellant's arguments are moot because the examiner is unclear as to what reference "Dogl" refers to. In contrast, the prior art of Win, Faybishenko, and Chang have been applied to reject the claims of the instant application.

Arguments (2):

I) Regarding Independent Claims 10, 20, and 27, Appellant argues that "the cited passage teaches monitoring search traffic generally and, based on the monitored information, displaying the results and number of the searches. However, Faybishenko fails to teach the claimed limitation of collecting runtime metadata relating to one or more result fields in a query statement, wherein the one or more result fields specify one or more data fields for which data is requested to be returned upon execution of the query statement". That is, while the cited passages from Faybishenko teach that a submitted query passing across a network adapter (i.e., the QRP adapter) may be monitored and/or logged, such a teaching pertains the query statement itself, rather than "metadata relating to one or more result fields in a query statement." Thus, even if, as the Examiner contends, Faybishenko teaches "logging an entire query...[where] the result fields are encompassed within that query," such a teaching still fails to teach the claimed limitation of "collecting runtime metadata relating to one or more result fields in a query statement"".

However, the examiner wishes to refer to Paragraph 112 of Faybishenko which states "In one embodiment a QRP adapter may monitor or log queries, results, number of hits, searches, results, etc. or generally the information passing through the QRP adapter. In one embodiment, a user interface may be provided through which providers may view the results of searches and hits performed by consumers--e.g. how many searches resulted in their entry being returned, how many users clicked through, etc. In one embodiment, a user interface may be provided through which providers may monitor and/or control the number of queries sent to them and also to throttle traffic (e.g. turn it off) if necessary. In some embodiments, a QRP interface may be able to access a registration file, for example to read at least part of the registration document or to write to replace or to add to at least part of the registration document" (Paragraph 112). The examiner further wishes to state that the independent claims define result fields as "wherein the one or more result fields specify one or more data fields for which data is requested to be returned upon execution of the query statement". The examiner further wishes to state that it is clear that because Faybishenko logs/records the entire query, result fields to be requested in that query are also stored. In addition, the "results" and "number of hits" are also stored, and thus, it is clear, contrary to appellant's assertions, that result fields of a query are stored in Faybishenko because the returned hits are also collected.

<u>Arguments (3):</u>

I) Regarding Independent Claims 10, 20, and 27, Appellant argues that "the current claim also recited "collecting runtime metadata…wherein the runtime metadata is collected after composition of the query statement, and wherein the runtime metadata is collected before the query statement is submitted for execution". Thus, in other words, the "collecting…" step of the current claim occurs at a moment in time after the query has been composed, but before the query has been submitted for execution. The Examiner cites to two references as teaching such a period of time: Faybishenko for teaching the limitation of "collecting runtime metadata…after composition of the query statement," and Chang for teaching the limitation of "collecting runtime metadata…before the query statement is submitted for execution." As discussed above...while the logging of Faybishenko may occur after query composition, such logging also occurs after the query has been submitted for execution. That is, because Faybishenko teaches monitoring a query that is passing through a network adapter, such monitoring occurs after the query has been submitted for execution. As such, the monitoring and logging taught by Faybishenko occur after both composition and submission of the query. Nonetheless, the Examiner argues that Chang teaches the claimed limitation of “collecting runtime metadata…before the query statement is submitted for execution,” More specifically, the Examiner states: “Change records, saves, and displays all concurrent typed keyboard strokes by a user. Because Chang records the typed keyboard stroked as a user is typing them, then as a result, the typed keywords are saved before a 'submission of execution."" As an initial note, Appellants note that Change does not teach "collecting runtime metadata relating to one or more result fields in a query statement" at any point in time, but merely teaches recording and displaying keyboard strokes as a user types them. Nonetheless, any “collecting” taught by Chang would occur not only before submission of a query, but before composition of the query as well. At best, Chang would merely teach recording the text a user types during query composition”.

However, the secondary reference of Faybishenko teaches the claimed collecting of runtime metadata relating to one or more result fields in a query statement. Moreover, the examiner wishes to refer to Column 5 of Chang which states “The logical operations of the executable begin at signal operation 202 where the CPU 104 awaits a user activity signal that is provided from the logical operations of FIG. 3. The user activity signal specifies when a particular key on the keyboard has been pressed by the user and when a particular mouse button has been clicked by the user. Query operation 204 detects whether the signal has been received. If not, then signal operation 202 continues to await the user activity signal. If the signal of user activity has been received, then query operation 206 detects whether the signal specifies a mouse click as opposed to a keyboard type. When a keyboard type is detected instead of a mouse click, then record operation 208 records the keyboard type as a user-driven event. As discussed, recording the user-driven event may involve one or more techniques, such as displaying a textual description of the keyboard type on the display screen, saving a description to an electronic file, and/or printing the textual description via a printer. Each character that is typed may be placed on the same line of the textual description as shown in the screenshots until the typed key is an <Enter> key or unless there is a control key such as <Backspace> or <Ctrl>+"A". After each keystroke is recorded, operational flow returns to signal operation 202" (Column 5, lines 4-26). The examiner further wishes to state that Chang records, saves, and displays all concurrent typed keyboard strokes by a user. Because Chang records the typed keyboard strokes as a user is typing them, then, as a result, the typed keywords are saved before a "submission of execution". Moreover, Figure 7 of Chang clearly shows an interface 606 that stores typed words by a user before the depression of an "enter" key that signifies a submission action. Because the Chang stores currently types words by a user, a query would clearly be stored before any submission action. In addition, the claimed collection of metadata before the submission of a query is not a pertinent element of claimed invention. Rather, Faybishenko clearly collects metadata but does so, after the submission of a query statement. In the end, both Faybishenko and the instant claimed invention collect metadata. Thus, the collection of metadata before the submission of a query is not a pertinent element because at the end, metadata of the query is collected in both Faybishenko and the instant claimed invention.

<u>Arguments (4):</u>

I) Regarding Independent Claims 10, 20, and 27, Appellant argues that "the combination of Faybishenko and Chang teaches logging the text of a query statement before both the composition and submission of the query statement, or after both composition and submission of the query statement. Thus, the combination of the cited references would at best teach collecting metadata before or after the claimed period of time (i.e., after composition of the query statement but before the query statement is submitted for execution), but fails to teach collecting metadata during the claimed period of time itself".

However, the examiner wishes to state that it is clear that Faybishenko stores queries after the composition of those queries (see "In one embodiment a QRP adapter may monitor or log queries, results, number of hits, searches, results, etc. or generally the information passing through the QRP adapter"). Moreover, because Chang stores typed keystrokes as a user is typing them (i.e. a keyword logger), then as a result, the subsequent statement is stored before a submission action, and this teaches the claimed time period of before submitting a query. Thus, the combination of Faybishenko's storage of a query after composition and Chang's storage of before a submission, teaches the aforementioned time period.

(11) Related Proceeding(s) Appendix

No decision rendered by a court or the Board is identified by the examiner in the Related Appeals and Interferences section of this examiner's answer.

For the above reasons, it is believed that the rejections should be sustained.

Respectfully submitted,

Mahesh Dwivedi
Patent Examiner
AU 2168
/Mahesh H Dwivedi/
Examiner, Art Unit 2168

/Tim T. Vo/
Supervisory Patent Examiner, Art Unit 2168

/Hosain T Alam/
Supervisory Patent Examiner, Art Unit 2166